

日 本 国 特 許 庁

PATENT OFFICE
JAPANESE GOVERNMENT

JC926 U.S. PT.
09/684328
10/10/00

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日

Date of Application:

1999年10月 7日

出 願 番 号

Application Number:

平成11年特許願第287268号

出 願 人

Applicant(s):

インターナショナル・ビジネス・マシーンズ・コーポレイション

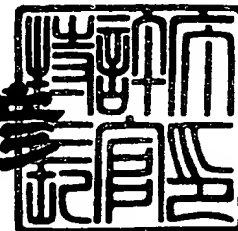


PRIORITY DOCUMENT
CERTIFIED COPY OF

2000年 3月10日

特許庁長官
Commissioner,
Patent Office

近 藤 隆 彦



出証番号 出証特2000-3014454

【書類名】 特許願

【整理番号】 JA999169

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 13/38

【発明者】

【住所又は居所】 神奈川県大和市下鶴間 1 6 2 3 番地 1 4 日本アイ・ピー・エム株式会社 東京基礎研究所内

【氏名】 小原 盛幹

【発明者】

【住所又は居所】 神奈川県大和市下鶴間 1 6 2 3 番地 1 4 日本アイ・ピー・エム株式会社 大和事業所内

【氏名】 古市 実裕

【特許出願人】

【識別番号】 390009531

【住所又は居所】 アメリカ合衆国 1 0 5 0 4、ニューヨーク州アーモンク
(番地なし)

【氏名又は名称】 インターナショナル・ビジネス・マシーンズ・コーポレーション

【代理人】

【識別番号】 100086243

【弁理士】

【氏名又は名称】 坂口 博

【連絡先】 0 4 6 2 - 7 3 - 3 3 1 8、3 3 2 5、3 4 5 5

【選任した代理人】

【識別番号】 100091568

【弁理士】

【氏名又は名称】 市位 嘉宏

【手数料の表示】

【予納台帳番号】 024154

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9304391

【包括委任状番号】 9304392

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 データ伝送装置およびその方法

【特許請求の範囲】

【請求項 1】

第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝送するデータ伝送装置であって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、

前記第 1 の装置は、

所定の条件を満たす前記伝送データをマージするマージ手段と、

所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成する伝送パケット生成手段と、

生成された前記伝送パケットを、前記第 2 の装置に伝送する伝送手段とを有するデータ伝送装置。

【請求項 2】

第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝送するデータ伝送方法であって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、

前記第 1 の装置において、

所定の条件を満たす前記伝送データをマージし、

所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成し、

生成された前記伝送パケットを、前記第 2 の装置に伝送するデータ伝送方法。

【請求項 3】

第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝送するプログラムであって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、

前記第 1 の装置において、

所定の条件を満たす前記伝送データをマージするマージステップと、
所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成しパケット生成ステップと、
生成された前記伝送パケットを、前記第 2 の装置に伝送する伝送ステップと
をコンピュータに実行させるプログラムを媒介する媒体。

【発明の詳細な説明】

【0 0 0 1】

【産業上の利用分野】

本発明は、画像等のデータを効率よく伝送するデータ伝送装置およびその方法に関する。

【0 0 0 2】

【従来の技術】

現在、コンピュータから表示装置に画像データを送るために、ラスターインターフェースが用いられる。このラスターインターフェースでは、全ての画素のデータがコンピュータから表示装置に送られるので、データ伝送量が多い。したがって、コンピュータに超高解像度の表示装置を接続するような場合、ラスターインターフェースで画像データを転送しようとする、コンピュータと表示装置とを接続する通信路のデータ伝送容量が足りなくなる可能性がある。

【0 0 0 3】

【発明が解決しようとする課題】

本発明は、上述した従来技術の問題点に鑑みてなされたものであり、例えば、コンピュータに超高解像度の表示装置を接続する場合に、画像データを効率よく伝送することができるデータ伝送装置およびその方法を提供することを目的とする。

【0 0 0 4】

【課題を達成するための手段】

上記目的を達成するために、本発明に係るデータ伝送装置は、第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝

送するデータ伝送装置であって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、前記第 1 の装置は、所定の条件を満たす前記伝送データをマージするマージ手段と、所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成する伝送パケット生成手段と、生成された前記伝送パケットを、前記第 2 の装置に伝送する伝送手段とを有する。

【0005】

また、本発明に係るデータ伝送方法は、第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝送するデータ伝送方法であって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、前記第 1 の装置において、所定の条件を満たす前記伝送データをマージし、所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成し、生成された前記伝送パケットを、前記第 2 の装置に伝送する。

【0006】

また、本発明にかかる媒体は、第 1 の装置から第 2 の装置に、伝送の対象となる伝送データを 1 つ以上、含む伝送パケットを伝送するプログラムであって、前記伝送データそれぞれは、予め指定された領域に対する処理を示す命令を含み、前記第 1 の装置において、所定の条件を満たす前記伝送データをマージするマージステップと、所定の範囲のデータ量の 1 つ上の前記伝送データおよび 1 つ以上の前記マージされた伝送データまたはこれらのいずれかを含む伝送パケットを生成しパケット生成ステップと、生成された前記伝送パケットを、前記第 2 の装置に伝送する伝送ステップとをコンピュータに実行させる。

【0007】

【発明の実施の形態】

以下、本発明の実施形態を説明する。

【0008】

〔画像処理装置 1 の構成〕

図 1 は、本発明に係る画像処理方法を実現する画像処理装置 1 の構成を示す図

である。

図1に示すように、画像処理装置1は、コンピュータ10および表示装置16が、通信路22を介して接続されて構成される。

【0009】

コンピュータ10は、汎用CPUおよびその周辺回路等を含むCPU102、RAMおよびその周辺回路等を含む主記憶装置104、ハードディスク装置およびCD-ROM装置などの二次記憶装置112、ビデオコントローラ12、フレームメモリ114および符号制御装置14が、システムバス100、ブリッジ106およびIOバス110を介して接続されて構成される。

つまり、コンピュータ10は、画像処理機能を有する一般的なコンピュータとしての構成を有している。

表示装置16は、符号制御装置18、表示制御装置20、フレームメモリ162および表示デバイス160から構成される。

【0010】

[コンピュータ10の構成]

図2は、図1に示したビデオコントローラ12の構成を例示する図である。

ビデオコントローラ12は、例えば図2に示すように、IOバスコントローラ120、描画エンジン122、フレームメモリコントローラ124および通信制御装置126から構成される。

ビデオコントローラ12は、これらの構成要素により、CPU102からの描画命令に応じて描画処理を行い、結果として得られた画像データをフレームメモリ114に記憶する。

また、ビデオコントローラ12は、表示制御装置16との間で描画命令等を伝送する。

【0011】

フレームメモリコントローラ124（図2）は、IOバスコントローラ120、描画エンジン122および通信制御装置126の間のメモリアクセス要求を調停、および、フレームメモリコントローラ124のリフレッシュ制御等を行う。

【0012】

I Oバスコントローラ 1 2 0 は、I Oバス 1 1 0 とビデオコントローラ 1 2 の構成部分とのインターフェースを行う。

【 0 0 1 3 】

描画エンジン 1 2 2 は、C P U 1 0 2 からの描画命令に応じて画像データを生成し、フレームメモリ 1 1 4 に記憶する。描画命令には、ビットブリット、直線・曲線の描画、フォントの展開などが含まれる。

【 0 0 1 4 】

通信制御装置 1 2 6 は、C P U 1 0 2 から描画命令を受け取り、符号制御装置 1 4 を介して表示装置 1 6 に送る。また、描画命令がフレームメモリ内の画素情報を含む場合は、通信制御装置 1 2 6 は、フレームメモリ 1 1 4 から画素データ（パターンデータおよびフォントデータ等）を読み出し、符号制御装置 1 4 に送る。

【 0 0 1 5 】

フレームメモリ 1 1 4 （図 1 ）は、表示装置 1 6 が表示する画像データおよび画素データを記憶する。なお、フレームメモリ 1 1 4 において、パターンデータ等の画素データを記憶する領域は、オフスクリーンメモリ領域とも呼ばれる。

【 0 0 1 6 】

図 3 は、図 1 に示したコンピュータ 1 0 側の符号制御装置 1 4 の構成を例示する図である。

図 3 に例示するように、符号制御装置 1 4 は、ビデオエンコーダ 1 4 0 および D D C マスタコントローラから構成される。

符号制御装置 1 4 は、これらの構成部分により、ビデオコントローラ 1 2 から受け取った画像データを符号化し、表示装置 1 6 に伝送する。

また、符号制御装置 1 4 は、表示装置 1 6 から描画命令のエラーデータ等（D D C ）を受信し、ビデオコントローラ 1 2 に対して出力する。

【 0 0 1 7 】

ビデオエンコーダ 1 4 0 は、ビデオコントローラ 1 2 の通信制御装置 1 2 6 （図 2 ）から受けた描画命令を通信路 2 2 を介して表示装置 1 6 に送信する。

【 0 0 1 8 】

DDCマスタコントローラ 1 4 2 は、表示装置 1 6 との間でエラーデータ等の DDCデータを双方向に伝送する。

【 0 0 1 9 】

〔表示装置 1 6 の構成〕

図 4 は、図 1 に示した表示装置 1 6 側の符号制御装置 1 8 の構成を例示する図である。

図 4 に例示するように、表示装置 1 6 の符号制御装置 1 8 は、ビデオデコーダ 1 8 0 および DDCスレーブコントローラ 1 8 2 から構成される。

符号制御装置 1 8 は、コンピュータ 1 0 の符号制御装置 1 4 に対応し、コンピュータ 1 0 から受けたデータを、表示制御装置 2 0 に対して出力する。

【 0 0 2 0 】

ビデオデコーダ 1 8 0 は、コンピュータ 1 0 のビデオエンコーダ 1 4 0 (図 3) に対応した動作を行い、コンピュータ 1 0 から画像データ等を受け、表示制御装置 2 0 に対して出力する。

【 0 0 2 1 】

DDCスレーブコントローラ 1 8 2 は、コンピュータ 1 0 の DDCマスタコントローラ 1 4 2 (図 3) に対応する動作を行い、コンピュータ 1 0 との間で DDCデータを伝送する。

【 0 0 2 2 】

表示デバイス 1 6 0 は、CRT表示装置、液晶表示装置あるいはプラズマディスプレイ等の表示装置であって、画像処理装置 1 のユーザに対して表示制御装置 2 0 から入力された画像データを視覚的に表示する。

【 0 0 2 3 】

図 5 は、図 1 に示した表示制御装置 2 0 の構成を示す図である。

図 5 に示すように、表示装置 1 6 の表示制御装置 2 0 は、通信制御装置 2 0 0 、描画エンジン 2 0 2 、フレームメモリコントローラ 2 0 4 および表示デバイスコントローラ 2 0 6 から構成される。

表示制御装置 2 0 は、これらの構成部分により、コンピュータ 1 0 からの描画命令に応じて描画処理を実行し、表示装置内のフレームメモリに生成した画像デ

ータを記憶する。

また、表示制御装置 2 0 は、表示デバイス 1 6 0 の種類に応じた所定の周期（6 0 Hz～8 5 Hz程度）で、表示デバイス 1 6 0 の表示をリフレッシュする。

【0 0 2 4】

フレームメモリコントローラ 2 0 4 は、コンピュータ 1 0 のフレームメモリコントローラ 1 2 4（図 2）と同様に、描画エンジン 2 0 2 および表示デバイスコントローラ 2 0 6 の間のメモリアクセス要求の調停、および、フレームメモリ 1 6 2 のリフレッシュ制御等を行う。

【0 0 2 5】

表示デバイスコントローラ 2 0 6 は、一定周期で全表示画素の画素データをフレームメモリ 1 6 2 から読み出し、表示デバイスに転送し、表示デバイス 1 6 0 に表示された画像をリフレッシュする。

【0 0 2 6】

描画エンジン 2 0 2 は、符号制御装置 1 8 から受けた描画命令に応じて描画処理を行い、生成した画像データをフレームメモリ 1 6 2 に記憶する。

【0 0 2 7】

通信制御装置 2 0 0 は、コンピュータ 1 0 との間で描画命令等を送受信する。

【0 0 2 8】

フレームメモリ 1 6 2 は、コンピュータ 1 0 のフレームメモリ 1 1 4 に記憶された画像データのコピーを記憶する。つまり、フレームメモリ 1 6 2 は、表示装置 1 6 が表示する画像データおよび画素データ（パターンデータおよびフォントデータ等）を記憶する。

【0 0 2 9】

本発明で重要な機能をもつ構成要素に、ビデオコントローラ（ハードウェア）、表示制御装置（ハードウェア）とグラフィックドライバ（ソフトウェア）がある。図 1 の構成では、ビデオコントローラと表示制御装置はハードウェアによる描画機能を持ち、グラフィックドライバが各描画命令の管理を行うと仮定している。本発明は、これらの 3 構成要素の実現がハードウェアかソフトウェアかに依存しない。

例えば、デバイスドライバが行う描画命令の管理をビデオコントローラがハードウェアにより行うことも可能である。また逆に、ビデオコントローラをソフトウェアで実現することも可能である。後者の場合は、主記憶内にフレームメモリが割り当てられ、CPUが主記憶上に描画を行う。この場合、コンピュータ内の符号制御装置は、主記憶上の画像情報や描画命令を表示装置に送る。

【0030】

[ソフトウェア構成]

図6は、本発明に係る画像処理プログラム3の構成を示す図である。

図6に示すように、画像処理プログラム3は、アプリケーションプログラム30(300-1~300-N)、オペレーティングシステム(OS)32、グラフィクスドライバ34、タイマ36、通信コントローラ40から構成される。

画像処理プログラム3は、例えば、記録媒体38(図1)に記録されて、あるいは、ネットワーク(図示せず)経由でコンピュータ10の二次記憶装置112に供給され、二次記憶装置112から主記憶104にロードされて実行され、本発明にかかる画像処理方法を実現する。

【0031】

[アプリケーションプログラム30(300-1~300-N)]

アプリケーションプログラム300-1~300-N(アプリケーションプログラム300-1~300-Nのいずれかを特定せずに示す場合には単にアプリケーションプログラム30とも記す)はそれぞれ、例えば画像データの生成を行う画像処理プログラムであって、生成した画像データを表示装置16(図1)に表示するグラフィクスAPI(Application Program Interface)により、OS32に対して描画処理を要求し、画像データをフレームメモリ114に記憶させる。

また、アプリケーションプログラム30は、同様に、グラフィクスAPIにより、OS32に対して画像データの読み出し要求を出し、フレームメモリ114から画像データを読み出させる。

【0032】

アプリケーションプログラム30が処理対象とするデータは、その処理内容に

応じて様々であり、一意には定まらず、例えば、アプリケーションプログラム 30 が処理対象とするデータは、アプリケーションプログラム 30 が静止画像の表示を行う場合には静止画像データであり、アプリケーションプログラム 30 が動画像を表示する場合には、処理対象のデータは M P E G 画像データ等であり、アプリケーションプログラム 30 が文字表示を行う場合には処理対象データはテキストデータである。

【 0 0 3 3 】

図 7 は、図 6 に示したアプリケーションプログラム 30 が O S 3 2 に対して発行するグラフィクス A P I のデータフォーマットを例示する図である。

アプリケーションプログラム 30 の処理結果は、O S 3 2 に対してグラフィクス A P I として出力され、グラフィクス A P I の内容は、O S 3 2 の種類等に依存する。例えば、アプリケーションプログラム 30 がグラフィクス A P I として W i n 3 2 システムの B i t B l t A P I を出力する場合には、図 7 に示すように、グラフィクス A P I には、描画領域の属性（描画先がスクリーンなのか、主記憶中のバッファなのか等を指示する）、描画先座標（描画領域矩形の左上隅の座標等を示す）、描画領域サイズ（描画領域矩形の幅・高さを示す）、ソース領域の属性、ソース領域の座標（ソース領域の矩形の左上隅の座標、但し、領域のサイズは描画領域サイズと同じなので、指定する必要はない）、および、R O P（ビット演算の演算方法を指示する）の各データが含まれる。

【 0 0 3 4 】

[O S 3 2]

O S 3 2 は、例えば、O S 2（I B M 社）あるいは W i n d o w s（マイクロソフト社）等のオペレーティングシステムである。

O S 3 2 は、アプリケーションプログラム 30 からグラフィクス A P I を受けて、グラフィックドライバ 34 に対して出力する。

なお、O S 3 2 は、アプリケーションプログラム 30 から受けたグラフィクス A P I を、必要に応じて、グラフィックドライバ 34 が処理可能な形式に変換して描画命令としてグラフィックドライバ 34 に対して出力する。

例えば、O S 3 2 がマイクロソフト社の W i n d o w s N T であり、アプリケ

ーションプログラム30からBitBlk(Bit Block Transfer) 命令をAPIとして受け取ったが、グラフィックスドライバ34がグラフィックスドライバ34がBitBlk命令をサポートしていないような場合に、OS32は、CopyBits命令を、グラフィックスドライバ34が実行可能な別の形式の命令(CopyBits命令等)に変換してグラフィックスドライバ34に対して出力する。

【0035】

また、OS32は、必要に応じて、描画命令のパラメータをグラフィックスドライバ34が処理可能な形式に変換してグラフィックスドライバ34に対して出力する。例えば、OS32は、描画処理の対象となる領域の一部が画面からはみ出す場合、あるいは、他のアプリケーションプログラム34のウィンドウに隠れる場合に、オペレーティングシステム32は、処理対象領域に対してクリッピング処理を行い、描画領域の座標(図7)を修正する。

【0036】

図8は、OS32(図6)がグラフィックスドライバ34に対して出力する描画命令を例示する図である。

図8に例示するように、OS32がグラフィックスドライバ34に対して描画命令としてBitBlk命令を発行する場合、このBitBlk命令には、アプリケーションプログラム30から入力されたグラフィクスAPIに含まれるデータ(描画領域の属性等)と、OS32がアプリケーションプログラム30からのデータを変更したり、新たに加えたデータ(マスクパターンの座標等)とが含まれる。

【0037】

【タイマ36】

タイマ36は、スケジューラ344および通信コントローラ40による操作に応じて、コンピュータ10から表示装置16に対して送信される描画命令それぞれ、あるいは、複数の描画命令を含むグループそれぞれに対応し、描画命令の送信から、表示装置16における処理の終了を示すACK信号の受信までの時間を監視するタイマ処理を行う。

【0038】

[命令解析ルーチン 3 4 0]

図 9 は、グラフィックスドライバ 3 4 (図 6) の命令解析ルーチン 3 4 0 の処理 (S 1 0) を示すフローチャートである。

グラフィックスドライバ 3 4 の命令解析ルーチン 3 4 0 は、OS 3 2 から入力された描画命令を解析し、解析結果に応じてグラフィックスドライバ 3 4 の他の構成部分の動作を制御する。

図 9 に示すように、ステップ 1 0 0 (S 1 0 0) において、命令解析ルーチン 3 4 0 は、OS 3 2 から受けた描画命令を付した関数 (IsExecutable 関数) をスケジューラ 3 4 4 に対して発行し、その描画命令が直ちに実行可能か否かをスケジューラ 3 4 4 に問い合わせる。スケジューラ 3 4 4 は、実行可能か否かを問い合わせられた描画命令が実行可能になると、その旨を命令解析ルーチン 3 4 0 に通知する。

描画命令の実行が可能である場合には S 1 0 2 の処理に進み、これ以外の場合には、実行可能になるまで S 1 0 0 の処理に留まる。

【0039】

ステップ 1 0 2 (S 1 0 2) において、命令解析ルーチン 3 4 0 は、描画命令を描画ルーチン 3 4 2 に対して出力し、描画ルーチン 3 4 2 を起動する。なお、描画ルーチン 3 4 2 は、命令解析ルーチン 3 4 0 から入力された描画命令に応じて描画処理を行い、描画処理の結果として得られた画像データをフレームメモリ 1 1 4 に書き込む。

【0040】

ステップ 1 0 4 (S 1 0 4) において、命令解析ルーチン 3 4 0 は、描画命令をスケジューラ 3 4 4 に関数 (EnQ 関数) として渡す。なお、この関数を受けたスケジューラ 3 4 4 は、表示装置 1 6 に描画命令を転送するなどの処理を行う。

【0041】

図 1 0 は、命令解析ルーチン 3 4 0 が処理結果として出力するデータを例示する図である。

図 1 0 に例示するように、命令解析ルーチン 3 4 0 は、OS 3 2 から入力され

た描画命令（図 8）それぞれに、命令の種類を示すヘッダを付して描画ルーチン 3 4 2 およびスケジューラ 3 4 4 に対して出力する。

図 1 0 に示した処理結果において、命令の種類は、スケジューラ 3 4 4 および表示装置 1 6（図 1）が処理する描画命令の種類を示し、オペレーティングシステム 3 2 から与えられる描画命令の種類とは異なる。

【 0 0 4 2 】

命令解析ルーチン 3 4 0 は、オペレーティングシステム 3 2 から与えられた描画命令を、スケジューラ 3 4 4 および表示装置 1 6 が処理可能な形式に変換して、これらの構成部分に対して出力する。

例えば、命令解析ルーチン 3 4 0 が、オペレーティングシステム 3 2 から `BitBlit` 命令を受け取った場合、命令解析ルーチン 3 4 0 は、ROP データ（図 7）を、`BitBlitSourceCopy`、`BitBlitPatternCopy` および `BitBlitDestinationInvert` など、さらに細かく分類する。

なお、命令解析ルーチン 3 4 0 が表示装置 1 6 がサポートしていない描画命令を受け取った場合には、命令解析ルーチン 3 4 0 は、描画命令をイメージ転送方式（`X←IMAGE`）の命令として指示する場合もある。

【 0 0 4 3 】

データの長さ（図 1 0）は、描画命令の種類ごとに異なるデータフォーマットのサイズを指定する。

命令固有のデータは、命令それぞれに固有のデータであって、例えば、OS 3 2 から描画命令として `BitBlit` 命令を受けた場合には、命令解析ルーチン 3 4 0 は、図 8 に例示したデータをそのまま命令固有のデータとする。

なお、以下に説明するスケジューラ 3 4 4 および通信コントローラ 4 0 が処理するデータは、図 1 0 に例示した命令解析ルーチン 3 4 0 と同じ形式を採る。

【 0 0 4 4 】

〔スケジューラ 3 4 4 〕

図 1 1 は、スケジューラ 3 4 4（図 6）の状態遷移を示す図である。

図 1 1 に示すように、スケジューラ 3 4 4 は、命令解析ルーチン 3 4 0 から描

画命令を受け取り、描画命令同士の間依存関係に基づいて適宜、実行順序を入れ替えたり、マージしたりする処理を行い、通信コントローラ 4 0 に対して処理の結果として得られた描画命令を出力する。

スケジューラ 3 4 4 において、描画命令は、以下の 3 つのいずれかの状態とされる。

【0 0 4 5】

【表 1】

(表 1) スケジューラ 3 4 4 における描画命令の状態：

状態 1 : P e n d i n g

既に表示装置 1 6 (図 1) に対して送信されているが、表示装置 1 6 から A C K 信号が返ってきていない状態。

状態 2 : R e a d y

他の描画命令と依存関係がなく、いつでも表示装置 1 6 に送信され得る状態。

状態 3 : D e p e n d e n t

他の描画命令と依存関係にあり、直ちに表示装置 1 6 に送信され得ない状態。

【0 0 4 6】

なお、表 1 に示した状態 3 (D e p e n d e n t 状態) は、以下のいずれかの条件が成立する描画命令の状態として定義され、D e p e n d e n t 状態にある描画命令は、依存関係のある他の描画命令が終了するまで、スケジューラ 3 4 4 から表示装置 1 6 (図 1) に送られることはない。

【0 0 4 7】

【表 2】

(表 2) D e p e n d e n t 状態の定義：

条件 1 - 1 :

・ 描画命令 (X ← I M A G E, X ← O P, X ← O P Z) の描画領域をオペランドとする未完了な描画命令 (Y ← O P X) が存在する。

条件 1 - 2 :

描画命令 (Y ← O P X) のオペランド X に描画する未完了な描画命令 (X ← I M A G E, X ← O P, X ← O P Z) が存在する。

【0048】

スケジューラ 344 は、例えば、描画命令同士がマージされたり、表示装置 16 からある描画命令に対する ACK 信号が返ってきたりして、描画命令間の依存関係が変化する可能性のある場合に、適宜、各描画命令の状態を変更する。

描画命令の状態を管理する方法としては、様々なものが考えられるが、その 1 つとして、上記 3 状態それぞれに対応する待ち行列を用いる状態の管理の方法が考えられる。

以下、この待ち行列を用いた状態管理方法を具体例として説明する。

スケジューラ 344 は、命令解析ルーチン 340 に対して、例えば下記 2 つの関数を提供する。

【0049】

【表 3】

(表 3) スケジューラ 344 が命令解析ルーチン 340 に用意する関数

関数 1-1: IsExecutable

オペレーティングシステム 32 からの描画命令が直ちに実行可能か調べる。

関数 1-2: EnQ

オペレーティングシステム 32 からの描画命令を、Ready 状態あるいは Dependent 状態としてスケジューラ 344 の管理下におく。

【0050】

なお、この待ち行列を用いる状態管理方法においては、スケジューラ 344 は、命令解析ルーチン 340 から関数に付されて入力された描画命令を、Ready Queue または Dependent Queue の末尾に追加する。

【0051】

関数 IsExecutable をさらに詳しく説明する。

命令解析ルーチン 340 から処理を要求されると、スケジューラ 344 の関数 IsExecutable は、OS 32 から命令解析ルーチン 340 に対して出された描画命令が実行可能であるか否かを判断し、判断結果 (TRUE/FALSE) を命令解析ルーチン 340 に返す (命令の実行可能性確認)。

【0052】

関数 `IsExecutable` は、下記 2 つの条件のいずれかが成立する場合には、描画命令を実行不可能と判断する。

この条件で関数 `IsExecutable` が描画命令を実行不可能と判断する理由は、例えば、この条件の下で、OS 32 からの描画命令 (`X ← IMAGE` 等) `IMAGE` など) を実行して、フレームメモリ 114 (図 1) に画像データを書き込むと、描画命令 `X' ← IMAGE` の実行結果が表示装置 16 のフレームメモリ 162 に正しく反映されず、結果的に描画命令 `Y ← OP X"` の実行が不正になるためである。

【0053】

但し、領域 `X`, `X'` は、同一領域の場合と一部が重複する場合とがある。

また、描画命令 `X' ← IMAGE` 自体は、その後の `X ← IMAGE` で上書きされるので、結果が不正でも構わない。

【0054】

【表 4】

(表 4) 描画命令が実行不可能と判断される条件：

条件 2-1：

その描画命令 (`X ← IMAGE`, `X ← OP`, `X ← OP Z`) の描画領域に描画する別の未完了な画素転送方式命令 (`X' ← IMAGE`, `X` と `X'` とは重複) が存在する。

条件 2-2：

命令 `X' ← IMAGE` より後に、その描画領域 (`X'`) をオペランドとする別の `Dependent` 状態の描画命令 (`Y ← OP X"`, `X'` と `X"` とは重複) が存在する。

【0055】

但し、表 4 に示した条件が成立する場合であっても、スケジューラ 344 が修正することを条件として、依存関係にある複数の描画命令が実行可能になる場合がある。

例えば、上記例において、関数 `IsExecutable` が、描画命令 `Y ← OP X"` を、画素転送方式の描画命令 `Y ← IMAGE` に修正すると、描画命令

X←IMAGEなどが実行可能になる。

【0056】

描画命令を修正するか否かの判断は、例えば、画像処理装置1（図1）においては、コンピュータ10と表示装置16との間の通信路22の伝送容量がシステムのボトルネックであると仮定し、通信路22上のデータ伝送量を減らせる場合のみ、IsExecutable関数がオペランドを伴う命令を画素転送方式の命令に修正する等の基準により行うことができる。

【0057】

また、描画命令が実行可能な場合、関数IsExecutableは、その命令がReady状態なのか、Dependent状態なのかを判断する。

描画命令がDependent状態であると判断した場合、関数IsExecutableは、その描画命令がいずれの描画命令と依存関係があるのかを判断し、判断結果をDependency Listに保存する。

また、関数IsExecutableは、描画命令が実行可能な場合で、その描画領域と重複する領域に描画する未完了な描画命令が存在する場合、古い命令をキャンセルして新しい命令とマージする。

【0058】

図12は、表3に示した関数IsExecutableの処理（S12）を示すフローチャートである。

ここまで説明した処理を実現するために、図12に示すように、ステップ120（S120）において、関数IsExecutableは、与えられた描画命令と同じ領域に描画する未完了の描画命令（X←IMAGE）が存在するか否かを判断し、存在しない場合にはマージ不可能（S122）と判断してS136の処理に進み、これ以外の場合にはマージ可能（S124）と判断してS126の処理に進む。

【0059】

ステップ126（S126）において、関数IsExecutableは、描画領域Xをオペランドとする別のDependent状態の描画命令（Y←OP X）が存在するか否かを判断し、存在する場合にはS128の処理に進み、こ

れ以外の場合には S 1 3 6 の処理に進む。

【0060】

ステップ 1 2 8 (S 1 2 8) において、関数 `IsExecutable` は、画素データを転送する描画命令 ($Y \leftarrow \text{IMAGE}$) に変更したことにより通信路 2 2 (図 1) に生じるデータ量の増加を、マージによって打ち消せるか否かを判断し、打ち消せる場合には S 1 3 2 の処理に進み、これ以外の場合には実行不可能と判断し (S 1 3 0)、処理を終了する。

【0061】

ステップ 1 3 2 (S 1 3 2) において、関数 `IsExecutable` は、描画命令 $Y \leftarrow \text{OP } X$ を、描画命令 $Y \leftarrow \text{IMAGE}$ に変換する。

【0062】

ステップ 1 3 4 (S 1 3 4) において、関数 `IsExecutable` は、S 1 3 2 の処理において選ばれた描画命令 $Y \leftarrow \text{IMAGE}$ の状態を、Ready 状態に変更する。

【0063】

ステップ 1 3 6 (S 1 3 6) において、関数 `IsExecutable` は、描画領域 X をオペランドとする別の描画命令 ($Y \leftarrow \text{OP } X$) があるか否かを判断し、ある場合には、描画命令 $X \leftarrow \text{OP } X$ を `Dependency List` に追加し (S 1 3 8)、これ以外の場合には S 1 4 0 の処理に進む。

【0064】

ステップ 1 4 0 (S 1 4 0) において、関数 `IsExecutable` は、オペランド Z を描画領域とする別の描画命令 ($Z \leftarrow$) があるか否かを判断し、ある場合には描画命令 $Z \leftarrow$ を `Dependency List` に加え (S 1 4 2)、これ以外の場合には S 1 4 4 の処理に進む。

【0065】

ステップ 1 4 4 (S 1 4 4) において、関数 `IsExecutable` は、マージ可能か否かを判断し、可能な場合にはこれらをマージ (S 1 4 8) し、これ以外の場合には処理を終了する。

【0066】

以下、関数 `EnQ` をさらに詳しく説明する。

スケジューラ 344 (図 6) の `EnQ` 関数は、命令解析ルーチン 340 から受け取った描画命令を、`Ready` 状態または `Dependent` 状態に設定し、それぞれの `Ready Queue` (図 11) または `Dependent Queue` の末尾に追加する。

【0067】

`EnQ` 関数は、`IsExecutable` 関数が `TURE` を命令解析ルーチン 340 に返した場合、つまり、`OS32` から入力された描画命令が、直ちに実行可能であると判断された場合に、命令解析ルーチン 340 により呼び出される。

【0068】

`Ready` 状態か `Dependent` 状態かの判断は、`IsExecutable` 関数により行われ、関数 `EnQ` は、この判断に基づいて、描画命令の状態を変更する。つまり、`EnQ` 関数は、描画命令の `Dependency List` が空でない場合、その描画命令を `Dependent` 状態とし、ある描画命令と依存関係のある描画命令がすべて終了するまで、その描画命令を `Ready` 状態にしない。

関数 `EnQ` は、一方、`Dependency List` が空の場合は、描画命令を直ちに表示装置 16 に転送することが可能なので、関数 `EnQ` は描画命令を `Ready` 状態とする。`Ready` 状態とされた描画命令は、関数 `DeQ` (表 5 等を参照して後述) により表示装置 16 に送られる。

【0069】

例えば、通信路 22 において、表示装置 16 からコンピュータ 10 方向へのデータ伝送 (`Uplink`) が遅くなるような場合には、描画命令が終了するたびに表示装置 16 からコンピュータ 10 側に `ACK` 信号を送信する方式を採ると、データ伝送が非効率になる可能性がある。このような場合には対処するためには、複数の描画命令に同じグループ ID を付与し、この描画命令のグループに含まれる複数の描画命令が、1 つの `ACK` 信号を共用する方法を採るとよい。関数 `EnQ` は、このような描画命令のグループ分けおよびグループ ID の付与を行う。

【0070】

なお、1つのグループのデータ転送量がある一定量を超えると、通信路22などでエラーが発生した場合の再送コストが増すので、関数EnQは、1つのグループのデータ量が所定の閾値を超えた場合に、最後の命令にEndOfGroupフラグを設定してグループを閉じ、それ以降にOS32から入力される描画命令を、次の新たなグループとし、新たなグループIDを付与する。

【0071】

図1.3は、表3に示した関数EnQの処理(S16)を示すフローチャートである。

図1.3に示すように、ステップ160(S160)において、関数EnQは、ある描画命令に、依存関係がある未完了の描画命令が存在するか否かを判断し、ある場合にはS162の処理に進み、これ以外の場合にはS164の処理に進む。

【0072】

ステップ162(S162)において、関数EnQは、描画命令をDependent Queueに追加する。

【0073】

ステップ164(S164)において、関数EnQは、その時点で使用しているグループIDを描画命令に割り振る。

【0074】

ステップ166(S166)において、関数EnQは、グループの総データ量が閾値Nより大きいのか否かを判断し、大きい場合にはS168の処理に進み、これ以外の場合にはS170の処理に進む。

【0075】

ステップ168(S168)において、関数EnQは、その時点で処理対象としているグループに新たな描画命令を加えないようにグループを閉じる。

【0076】

ステップ170(S170)において、関数EnQは、描画命令をReady Queue(図11)に加える。

【 0 0 7 7 】

また、スケジューラ 3 4 4 は、通信コントローラ 4 0 に対して、下記 2 つの関数を提供する。

【 0 0 7 8 】

【表 5】

(表 5) スケジューラ 3 4 4 が描画ルーチン 3 4 2 に用意する関数

関数 2 - 1 : D e Q

R e a d y 状態 (表 1) の描画命令を R e a d y Q u e u e (図 1 1) の先頭より 1 つ取り出し、通信コントローラ 3 4 2 に渡す。同時に、その描画命令を P e n d i n g 状態にして、P e n d i n g Q u e u e の末尾に移す。

関数 2 - 2 : A c k

表示装置 1 6 から A C K 信号が返ってきた場合に、スケジューラ 3 4 4 内の描画命令の依存関係を更新する。

【 0 0 7 9 】

以下、関数 D e Q をさらに詳細に説明する。

関数 D e Q は、R e a d y 状態の描画命令を古い順に (R e a d y Q u e u e の先頭から) 取り出し、通信コントローラ 4 0 に渡すとともに、渡した描画命令の状態を P e n d i n g 状態にする。

描画命令をグループにまとめてディスプレイへ転送する場合には、関数 D e Q は、R e a d y 状態の描画命令がなくなった時点でグループを閉じ、E n Q 関数と同様に、最後の命令に E n d O f G r o u p フラグを設定し、新たなグループ I D の設定を行う。

【 0 0 8 0 】

図 1 4 は、表 4 に示した関数 D e Q の処理 (S 1 8) を示すフローチャートである。

図 1 4 に示すように、ステップ 1 8 0 (S 1 8 0) において、関数 D e Q は、R e a d y Q u e u e (図 1 1) が空か否かを判断し、空の場合には処理を終了し、これ以外の場合には S 1 8 2 の処理に進む。

【 0 0 8 1 】

ステップ182 (S182) において、関数DeQは、Ready Queueから描画命令を1つ取り出す。

【0082】

ステップ184 (S184) において、関数DeQは、Ready Queueが空か否かを判断し、空の場合にはS186の処理に進み、これ以外の場合にhS188の処理に進む。

【0083】

ステップ186 (S186) において、関数DeQは、グループを閉じる。

【0084】

ステップ188 (S188) において、関数DeQは、描画命令をPending Queueに追加する。

【0085】

関数ACKは、ACK信号が表示装置16から返ってくると、受信したACK信号を返したグループに対応し、例えば、ACK信号を返したグループに付されたグループIDが付されたタイマ処理が稼働中ならば、それを停止する。

このタイマ処理は、表示装置16に送った描画命令のグループそれぞれについてACK信号が返ってこない場合のエラー処理をするために、描画命令が送信された直後に、通信コントローラ40により起動される。

【0086】

ここで、受信したACK信号に対応するタイマ処理が既に停止していた場合は、タイマイベントが発生し、エラー処理が開始されているので、関数ACKは何もしないで終了する。

タイマを停止すると、関数ACKは、ACK信号を返したグループの描画命令(Pending状態)を、スケジューラ344から削除する。

Pending状態の描画命令が削除されると、描画命令間の依存関係が解消されることがある。

関数ACKは、Dependent状態にあり、受信した信号ACKに対応する描画命令と依存関係にある描画命令を探し、探し出された描画命令の内、依存関係が解消されているものがあれば、その描画命令をReady状態にする。

【0087】

以下、関数ACKをさらに詳細に説明する。

図15は、図11に示した関数ACK (S20) の処理を示すフローチャートである。

図15に示すように、ステップ200 (S200) において、関数ACKは、表示装置16からACK信号を受信すると、このACK信号に対応するタイマ36 (図6) の処理が稼働中か否かを判断し、稼働中である場合にはS202の処理に進み、これ以外の場合にはS204の処理に進む。

【0088】

ステップ202 (S202) において、関数ACKは、S200の処理において受信したACK信号に対応するタイマ36の処理を停止する。

【0089】

ステップ204 (S204) において、関数ACKは、S200の処理において受信したACK信号に対応するグループに含まれる全ての描画命令を、Pending Queueから削除する。

【0090】

ステップ206 (S206) において、関数ACKは、S204の処理において削除した描画命令と依存関係を持つ描画命令のDependency Listから、S204の処理において削除した描画命令を削除する。

【0091】

ステップ208 (S208) において、S206の処理において描画命令を削除したDependency Listが空になったか否かを判断し、空になった場合にはS210の処理に進み、これ以外の場合には処理を終了する。

【0092】

ステップ210 (S210) において、関数ACKは、S208の処理において、Dependency Queueが空になったと判断された描画命令を、Dependent QueueからReady Queueに移す。

【0093】

さらに、スケジューラ344は、表示装置16からACK信号が帰ってこない

時のエラー処理をするために、エラーが発生した描画命令を、Pending Queue (図11) から取り出し、Pending 状態から Ready 状態に移行させ、Ready Queue の末尾に移すタイマイイベントハンドラ(timer event handler)関数 AckTimeout を提供する。

【0094】

関数 AckTimeout は、コンピュータ10から表示装置16に送られた描画命令が正常に終了せず、一定時間経過してもコンピュータにACK信号が返ってこない場合に、タイマ36によりタイマイイベントが発生することにより呼び出される。

関数 AckTimeout は、スケジューラ344内に残されている Ready 状態の描画命令のうち、最後のものに EndOfGroup フラグを設定し、グループIDの値をインクリメントしておく。

【0095】

この処理により、スケジューラ344は、エラーが発生したグループの描画命令を、新たなグループとして再登録し、さらに、エラーが発生したグループに含まれる描画命令を Pending 状態から Ready 状態に変える。

このように再登録されたグループの描画命令は、通信コントローラ40により表示装置16に対して再送信される。

【0096】

図16は、図11に示したタイマイイベントハンドラ AckTimeout の処理 (S22) を示すフローチャートである。

図16に示すように、ステップ220 (S220) において、関数 AckTimeout は、Ready Queue に残された描画命令のグループを閉じる。

さらに、関数 AckTimeout は、閉じたグループの最後の描画命令に EndOfGroup のフラグを設定し、グループIDをインクリメントする。

【0097】

ステップ222 (S222) において、関数 AckTimeout は、エラーが発生したグループの全ての描画命令に新たなグループIDを付し、Pende

ing Queue から Ready Queue に移す。

【0098】

〔描画ルーチン 342〕

描画ルーチン 342 (図 6) は、例えば、描画エンジン 122 (図 2) を用いて、命令解析ルーチン 340 から入力された描画命令の実行に必要なグラフィクス演算を行い、画像データを生成してフレームメモリ 114 に記憶する。

【0099】

〔通信コントローラ 40〕

通信コントローラ 40 (図 6) は、スケジューラ 340 から Ready 状態の描画命令を取り出す。

また、通信コントローラ 40 は、必要に応じてフレームメモリ 114 から画像データを読み込み、描画命令および画素データを含むパケットを作成し、通信制御装置 126 (図 2) を用いて、作成したパケットを表示装置 16 に転送する。

【0100】

なお、通信コントローラ 40 は、描画命令ごとに ACK 信号を受けるようにする場合には、パケットを転送するたびに、タイマ 36 を操作してタイムアウト監視用のタイマ処理を行わせる。

また、それぞれ複数の描画命令を含むグループ単位で ACK 信号を受けるようにする場合は、通信コントローラ 40 は、パケットを作成する際に描画命令に含まれる End Of Group フラグをチェックし、グループ最後の描画命令に対応するパケットを送信した後に、通信コントローラ 40 はタイマ 36 に対する操作を行う。

【0101】

図 17 は、図 6 に示した通信コントローラ 40 の処理 (S24) を示すフローチャートである。

図 17 に示すように、ステップ 240 (S240) において、通信コントローラ 40 は、スケジューラ 344 の Ready Queue (図 11) が空か否かを判断し、空の場合には S240 の処理に留まり、これ以外の場合には S242 の処理に進む。

【0102】

ステップ242 (S242) において、通信コントローラ40は、スケジューラ344のReady Queueから描画命令を取り出す。

【0103】

ステップ244 (S244) において、通信コントローラ40は、必要に応じてフレームメモリ114から画素データを読み出し、描画命令などを含むパケットを生成する。

【0104】

ステップ246 (S246) において、通信コントローラ40は、通信制御装置126を制御して、S244の処理において生成したパケットを、通信路22 (図1) を介して表示装置16に対して送信させる。

【0105】

ステップ248 (S248) において、通信コントローラ40は、S246の処理において表示装置16に送信したパケットに含まれる描画命令が、グループ最後の描画命令であるか否かを判断し、グループ最後の描画命令である場合にはS250の処理に進み、これ以外の場合にはS240の処理に戻る。

【0106】

ステップ250 (S250) において、通信コントローラ40は、タイマ36を操作して、S246の処理において送信したグループに対応するタイマ処理を起動させる。

【0107】

〔画像処理装置1の動作〕

以下、さらに図18および図19を参照して、画像処理装置1の動作を説明する。

図18は、画像処理装置1 (図1) の動作の内、アプリケーションプログラム30 (図5) から出された描画命令が、OS32などを介してスケジューラ344の待ち行列に入れられるまでの処理を示すシーケンス図である。

図19は、通信コントローラ40 (図6) が、スケジューラ344の待ち行列から描画命令を取り出し、表示装置16に送信する処理を示すシーケンス図であ

る。

【0 1 0 8】

図 1 8 に示すように、アプリケーションプログラム 3 0 が、描画 API を用いて、オペレーティングシステム 3 2 に対して描画要求を出す（要求 1）。

上記要求 1 を受けた OS 3 2 は、グラフィックスドライバ 3 4 を起動し、命令解析ルーチン 3 4 0 に対して描画命令を出す（要求 2）。

上記要求 2 を受けた命令解析ルーチン 3 4 0 は、スケジューラ 3 4 4 に対して、描画命令が直ちに実行可能か否かを問い合わせる（要求 3）。

上記要求 3 を受けたスケジューラ 3 4 4 は、描画命令が、直ちに実行可能か否かを調べ、命令解析ルーチン 3 4 0 に返答する（応答 4）。

【0 1 0 9】

上記応答 4 の処理において、スケジューラ 3 4 4 が描画命令が直ちに実行可能と判断し、その旨の応答をした場合、命令解析ルーチン 3 4 0 は、描画ルーチン 3 4 2 に対して描画命令を出す（要求 5）。

描画ルーチン 3 4 2 は、要求 5 の処理において受けた描画命令に従って描画処理を行い、生成した画像データをフレームメモリ 1 1 4 に記憶する。

描画処理が終了すると、描画ルーチン 3 4 2 は、命令解析ルーチン 3 4 0 に対して、描画処理が正常に終了したことを伝える（応答 6）。

【0 1 1 0】

応答 6 の処理において描画処理の政情終了の通知を受けた命令解析ルーチン 3 4 0 は、描画命令をスケジューラ 3 4 4 の待ち行列に加える（要求 7）。

スケジューラ 3 4 4 は、描画命令が正常に待ち行列に加えられたことを、命令解析ルーチンに返答する（応答 8）。

命令解析ルーチン 3 4 0 は、描画処理が終了したことをオペレーティングシステム 3 2 に伝える（応答 9）。

オペレーティングシステム 3 2 は、描画処理が終了したことをアプリケーションプログラム 3 0 に伝える（応答 1 0）。

【0 1 1 1】

さらに、図 1 9 に示すように、通信コントローラ 4 0 は、スケジューラ 3 4 4

の待ち行列から描画命令を取り出し、表示装置 16 に送信する。

つまり、通信コントローラ 40 は、スケジューラ 344 に対して、Ready 状態の描画命令を待ち行列の中から取り出すように要求する（要求 11）。

上記要求 11 に応じて、スケジューラ 344 は、通信コントローラ 40 に、Ready 状態の描画命令を一つ渡す（応答 12）。

【0112】

通信コントローラ 40 はパケットを作成し、表示装置 16 に送信する（要求 13）。この場合の描画とは、コンピュータ 10 のフレームメモリ 114 への描画処理を意味し、実際にディスプレイ 114 に表示されているわけではない。

【0113】

通信コントローラ 40 は、描画命令送信から ACK 信号が返るまでの時間を示すタイムアウト値を指定してタイマの起動を要求する（要求 14）。

タイマ 36 は、通信コントローラ 40 に対して、正常にタイマが起動されたことを伝える（応答 15）。

表示装置 16 は、描画処理が正常に終了したことを、通信コントローラ 40 に対して知らせる（応答 16）。

【0114】

通信コントローラ 40 は、上記応答 16 を受けると、スケジューラ 344 に対して、描画命令間の依存関係の変更を依頼する（要求 17）。

スケジューラ 344 は、描画命令間の依存関係を更新し、タイマ 36 のタイマ処理を停止する（要求 18）。

タイマ 36 は、スケジューラ 344 に対して、タイマ処理が正常に停止したことを伝える（応答 19）。

スケジューラ 344 は、通信コントローラ 40 に対して、上記要求 7 に関する処理が正常終了したことを伝える（応答 20）。

上記要求 4 の処理においてタイマを起動してから、一定時間（指定したタイムアウト値）を経過しても、応答 6 が返らない場合、タイマは、スケジューラに対して、描画命令の再送を要求する（要求 21）。

【0 1 1 5】

【発明の効果】

以上説明したように、本発明に係るデータ伝送装置およびその方法は、例えば、コンピュータに超高解像度の表示装置を接続する場合に、画像データを効率よく伝送することができる。

【図面の簡単な説明】

【図 1】

本発明に係る画像処理方法を実現する画像処理装置の構成を示す図である。

【図 2】

図 1 に示したビデオコントローラの構成を例示する図である。

【図 3】

図 1 に示したコンピュータ側の符号制御装置の構成を例示する図である。

【図 4】

図 1 に示した表示装置側の符号制御装置の構成を例示する図である。

【図 5】

図 1 に示した表示制御装置の構成を示す図である。

【図 6】

本発明に係る画像処理プログラムの構成を示す図である。

【図 7】

図 6 に示したアプリケーションプログラムが OS に対して発行するグラフィックス API のデータフォーマットを例示する図である。

【図 8】

OS (図 6) がグラフィックスドライバに対して出力する描画命令を例示する図である。

【図 9】

グラフィックスドライバ (図 6) の命令解析ルーチンの処理 (S 1 0) を示すフローチャートである。

【図 1 0】

命令解析ルーチンが処理結果として出力するデータを例示する図である。

【図 1 1】

スケジューラ（図 6）の状態遷移を示す図である。

【図 1 2】

表 3 に示した関数 `IsExecutable` の処理（S 1 2）を示すフローチャートである。

【図 1 3】

表 3 に示した関数 `EnQ` の処理（S 1 6）を示すフローチャートである。

【図 1 4】

表 4 に示した関数 `DeQ` の処理（S 1 8）を示すフローチャートである。

【図 1 5】

図 1 1 に示した関数 `ACK`（S 2 0）の処理を示すフローチャートである。

【図 1 6】

図 1 1 に示したタイマイイベントハンドラ `AckTimeout` の処理（S 2 2）を示すフローチャートである。

【図 1 7】

図 6 に示した通信コントローラの処理（S 2 4）を示すフローチャートである。

【図 1 8】

画像処理装置（図 1）の動作の内、アプリケーションプログラム（図 5）から出された描画命令が、OSなどを介してスケジューラの待ち行列に入れられるまでの処理を示すシーケンス図である。

【図 1 9】

通信コントローラ（図 6）が、スケジューラの待ち行列から描画命令を取り出し、表示装置に送信する処理を示すシーケンス図である。

【符号の説明】

1・・・画像処理装置

10・・・コンピュータ

100・・・システムバス

102・・・CPU

- 1 0 4 . . . 主記憶
- 1 0 6 . . . ブリッジ
- 1 1 0 . . . I O バス
- 1 1 2 . . . 二次記憶装置
- 1 1 4 . . . フレームメモリ
- 1 2 . . . ビデオコントローラ
 - 1 2 0 . . . I O バスコントローラ
 - 1 2 2 . . . 描画エンジン
 - 1 2 4 . . . フレームメモリコントローラ
 - 1 2 6 . . . 通信制御装置
- 1 4 . . . 符号制御装置
 - 1 4 0 . . . ビデオエンコーダ
 - 1 4 2 . . . DDC マスタコントローラ
- 1 6 . . . 表示装置
 - 1 6 0 . . . 表示デバイス
 - 1 6 2 . . . フレームメモリ
- 1 8 . . . 符号制御装置
 - 1 8 0 . . . ビデオデコーダ
 - 1 8 2 . . . DDC スレーブコントローラ
- 2 0 . . . 表示制御装置
 - 2 0 0 . . . 通信制御装置
 - 2 0 2 . . . 描画エンジン
 - 2 0 4 . . . フレームメモリコントローラ
 - 2 0 6 . . . 表示デバイスコントローラ
- 2 2 . . . 通信路
- 3 . . . 画像処理プログラム
 - 3 0 (3 0 0 - 1 ~ 3 0 0 - N) . . . アプリケーションプログラム
 - 3 2 . . . O S
 - 3 4 . . . グラフィックスドライバ

340・・・命令解析ルーチン

342・・・描画ルーチン

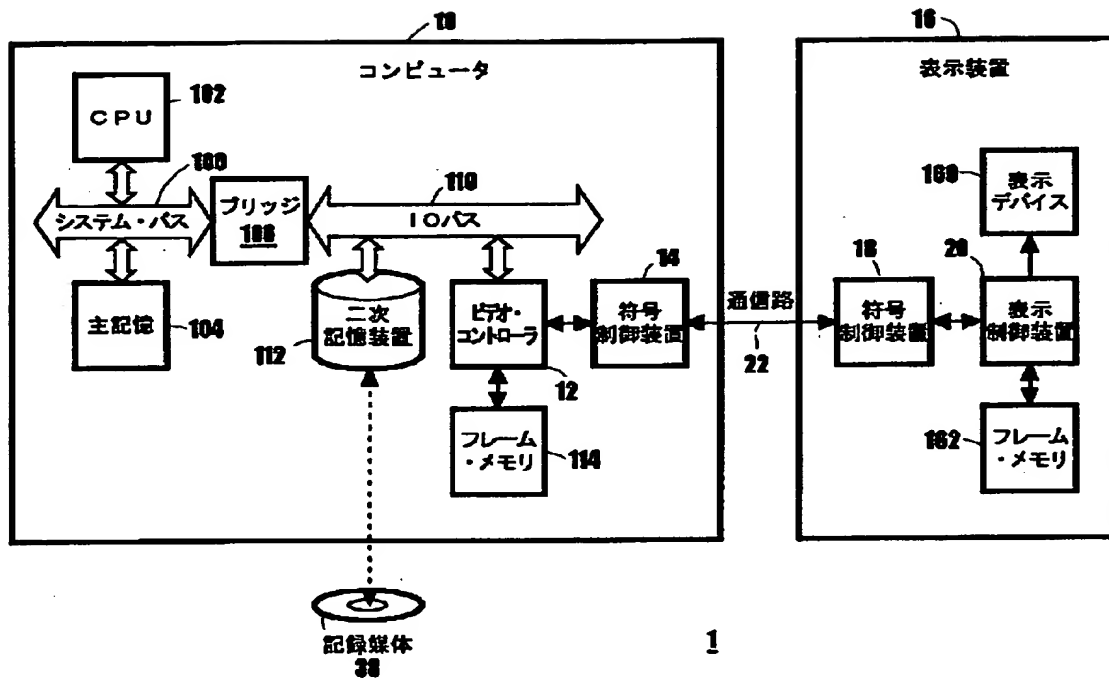
344・・・スケジューラ

36・・・タイマ

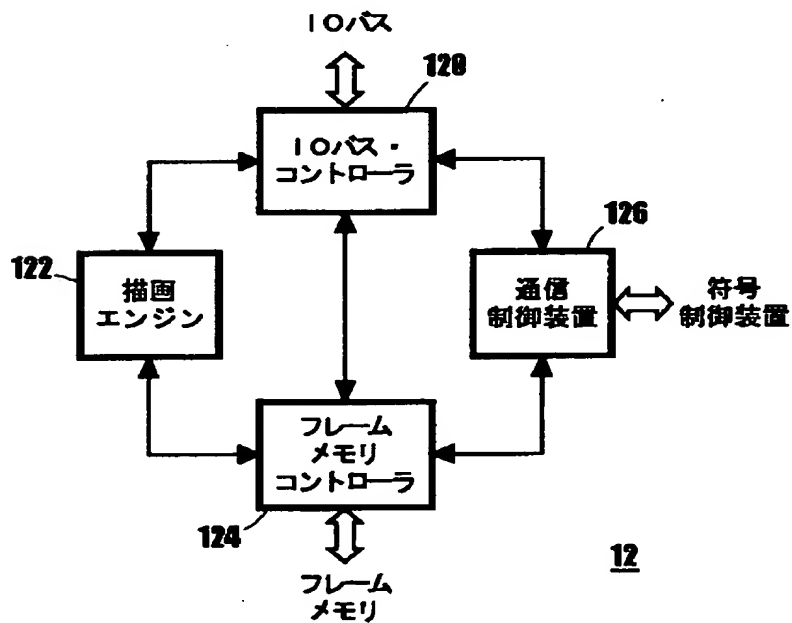
40・・・通信コントローラ

【書類名】 図面

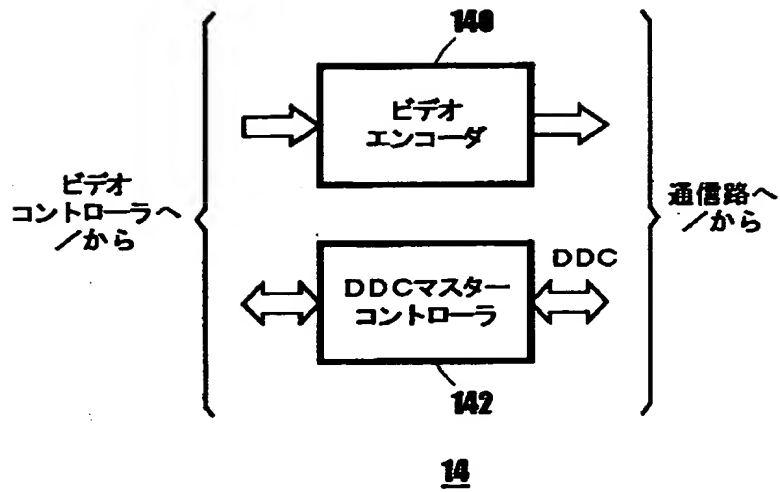
【図 1】



【図 2】



【図 3】

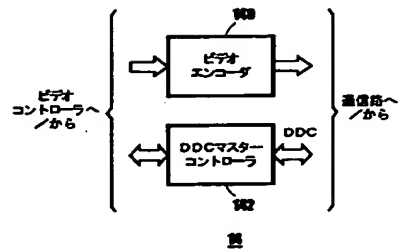


【図 4】

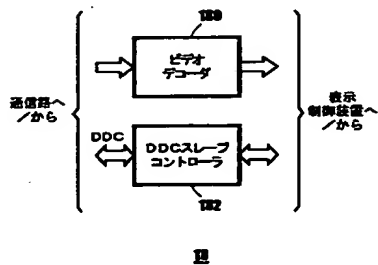
整理番号 JA999169

(2/13)

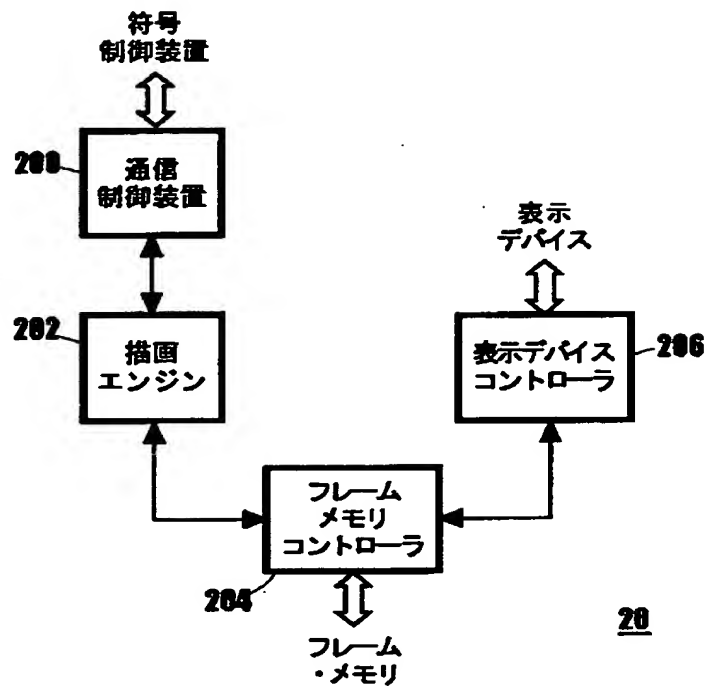
【図 3】



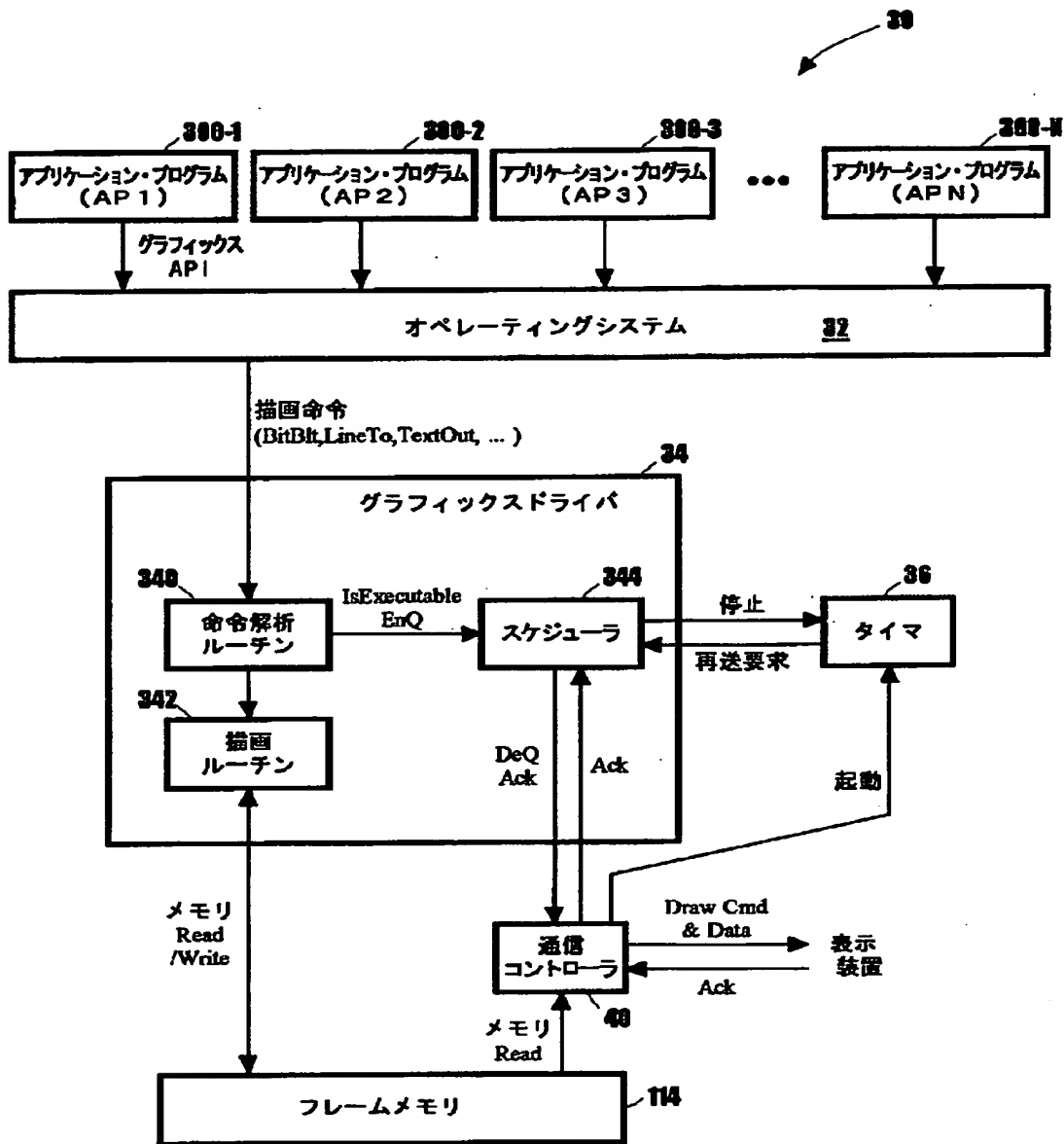
【図 4】



【図 5】



【図 6】



3

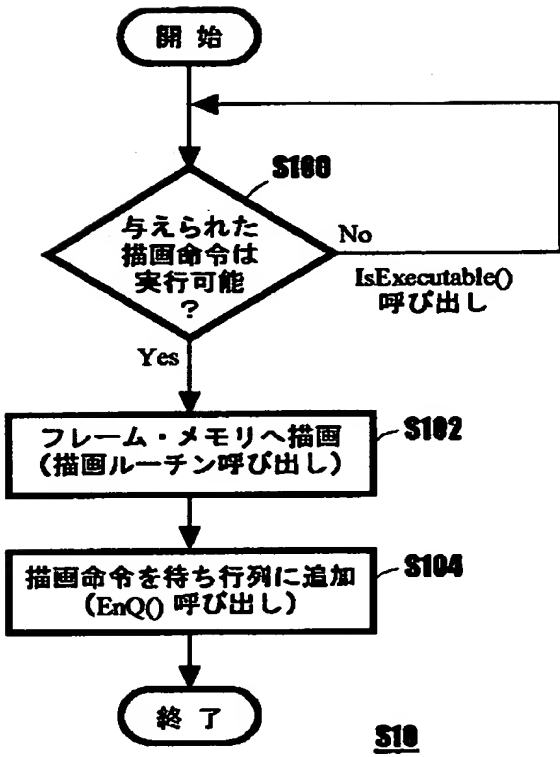
【図 7】

描画領域の 属性	描画先 座標	描画領域 サイズ	ソース領域の 属性	ソース領域の 座標	ROP
-------------	-----------	-------------	--------------	--------------	-----

【図 8】

描画領域の 属性	ソース領域の 属性	マスク・パターン の属性	クリッピング 情報	色 情報	...	描画先座標 とサイズ	マスク・ パターンの座標	パターン 属性	パターン 座標
-------------	--------------	-----------------	--------------	---------	-----	---------------	-----------------	------------	------------

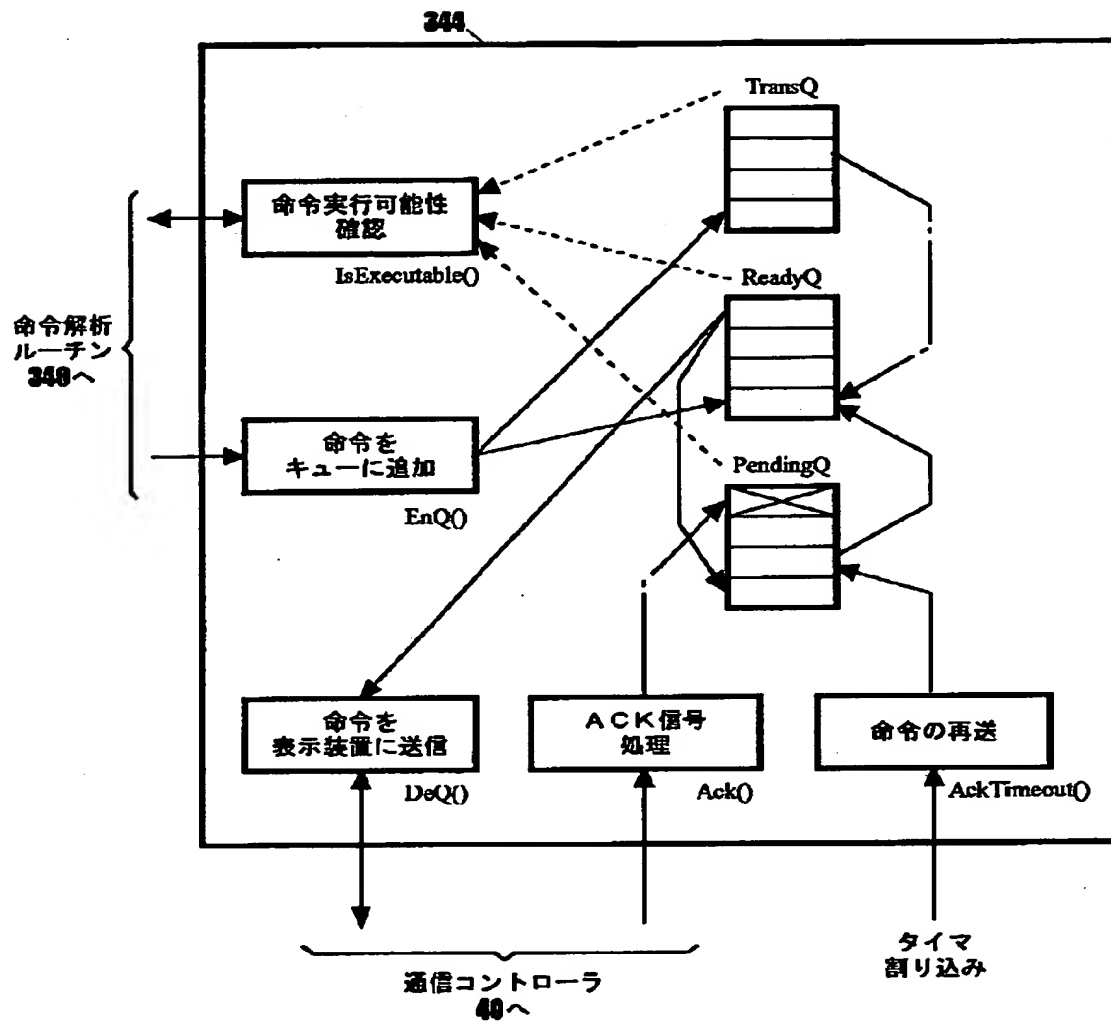
【図 9】



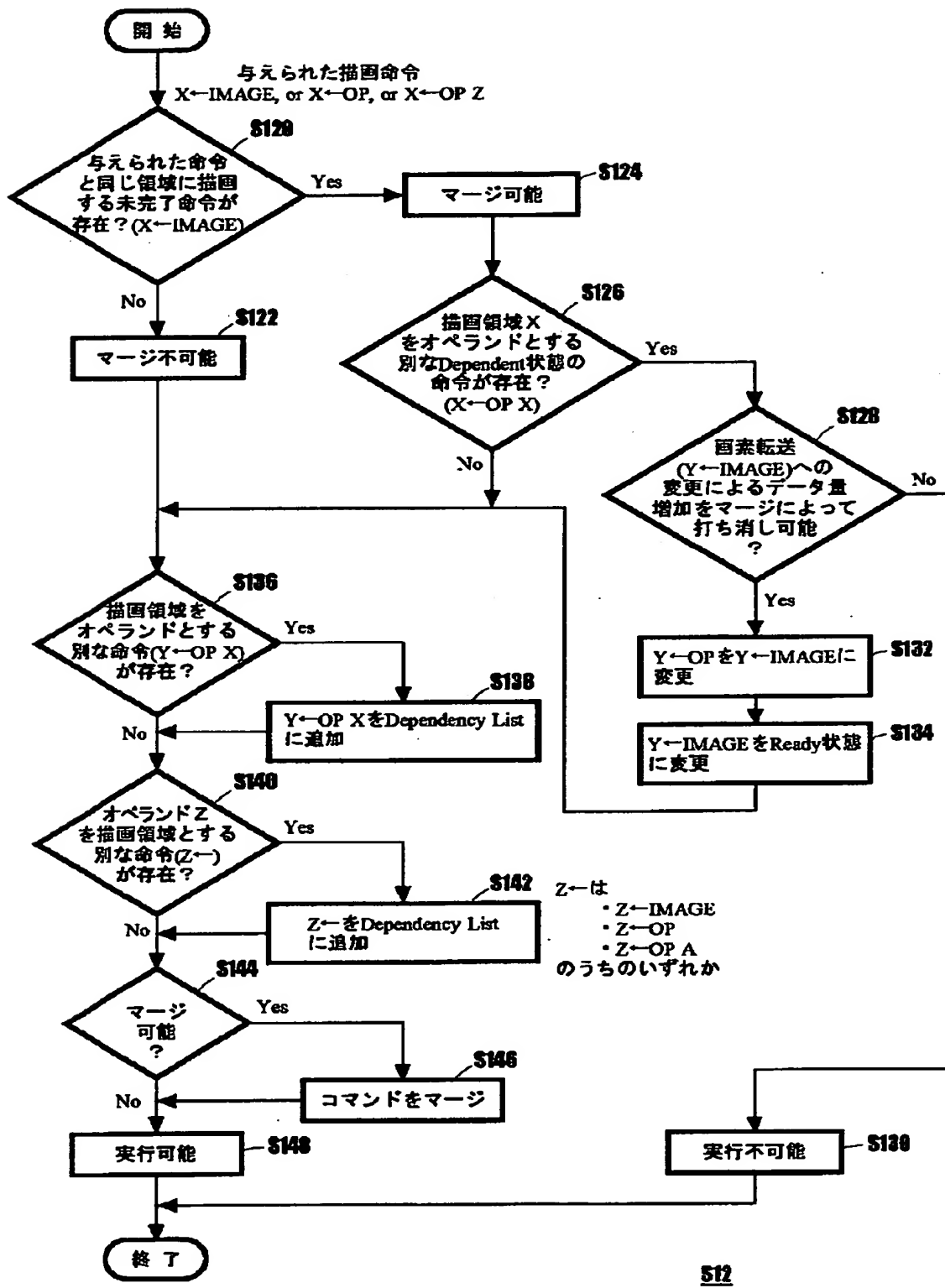
【図 10】

命令の種類	データの長さ	命令固有のデータ
-------	--------	----------

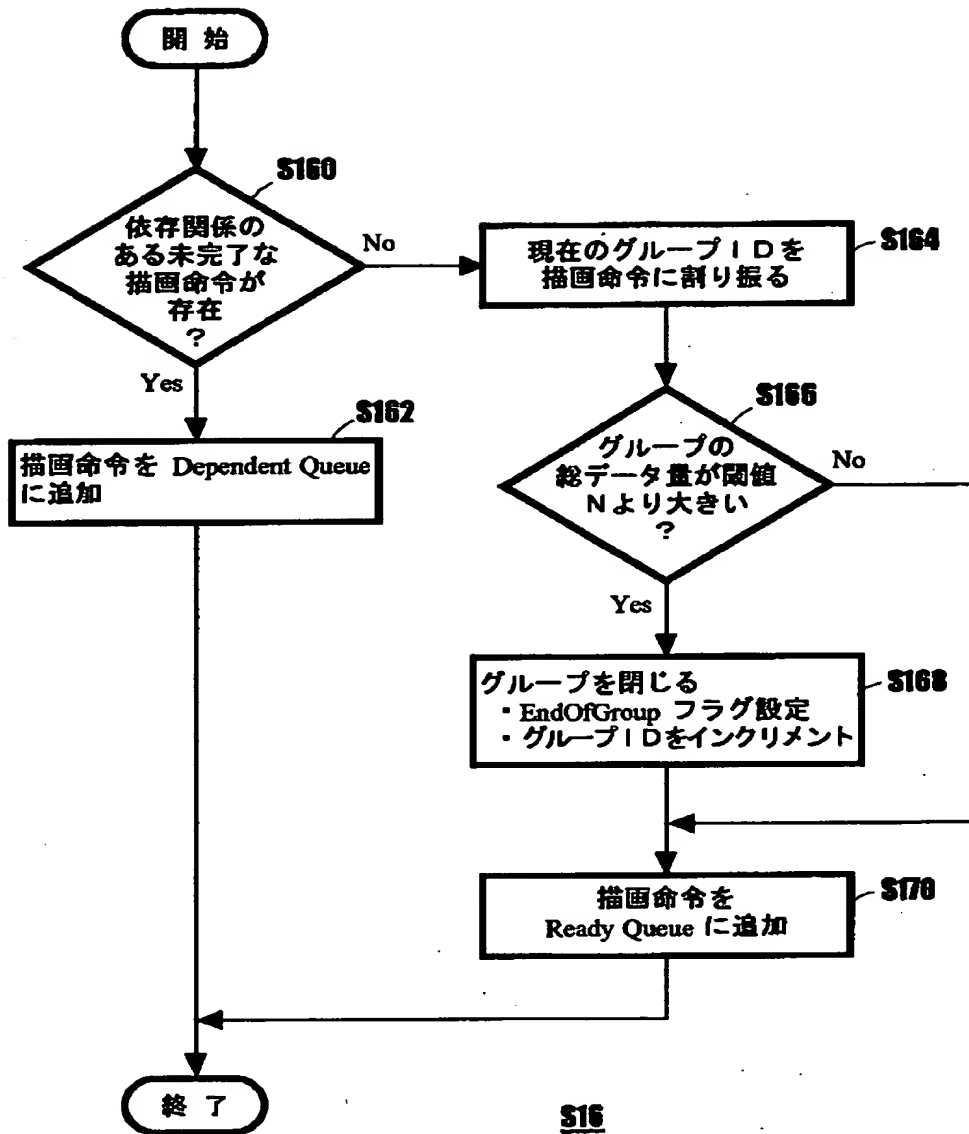
【図 11】



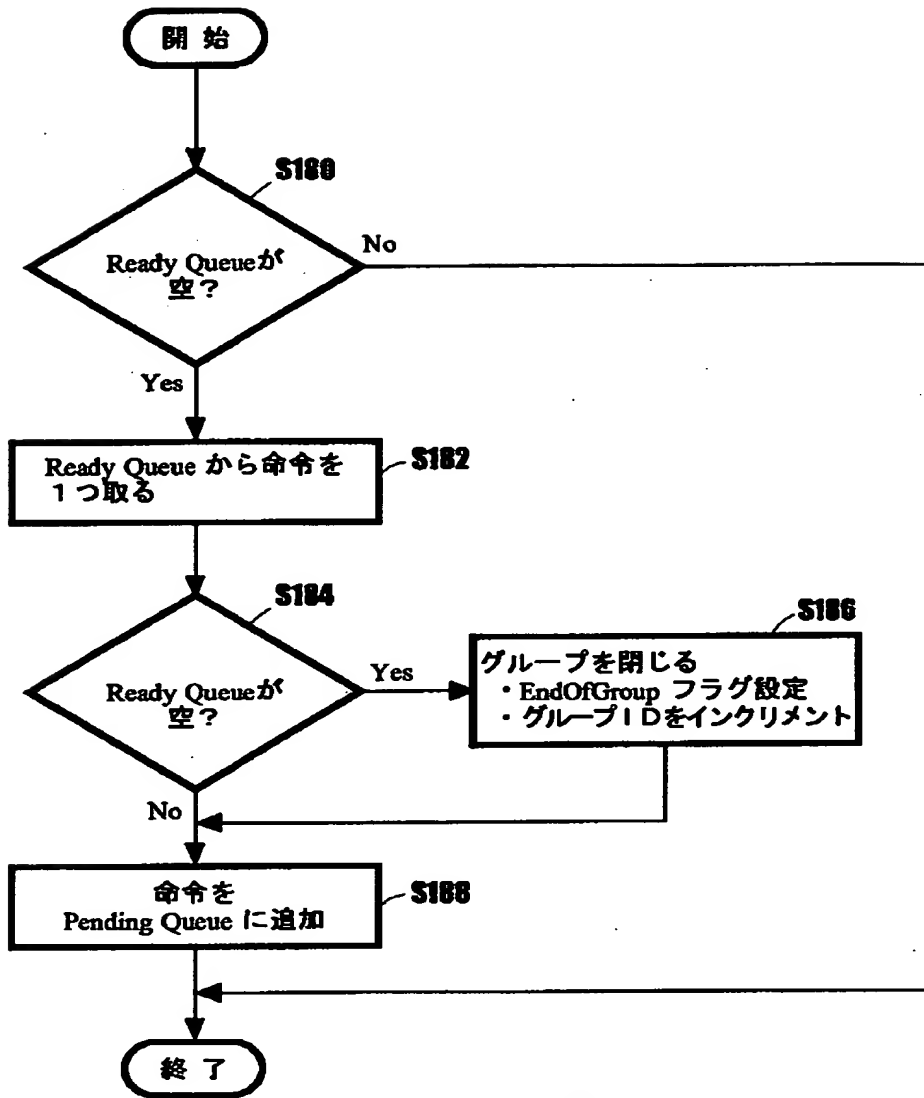
【図 12】



【図 1 3】

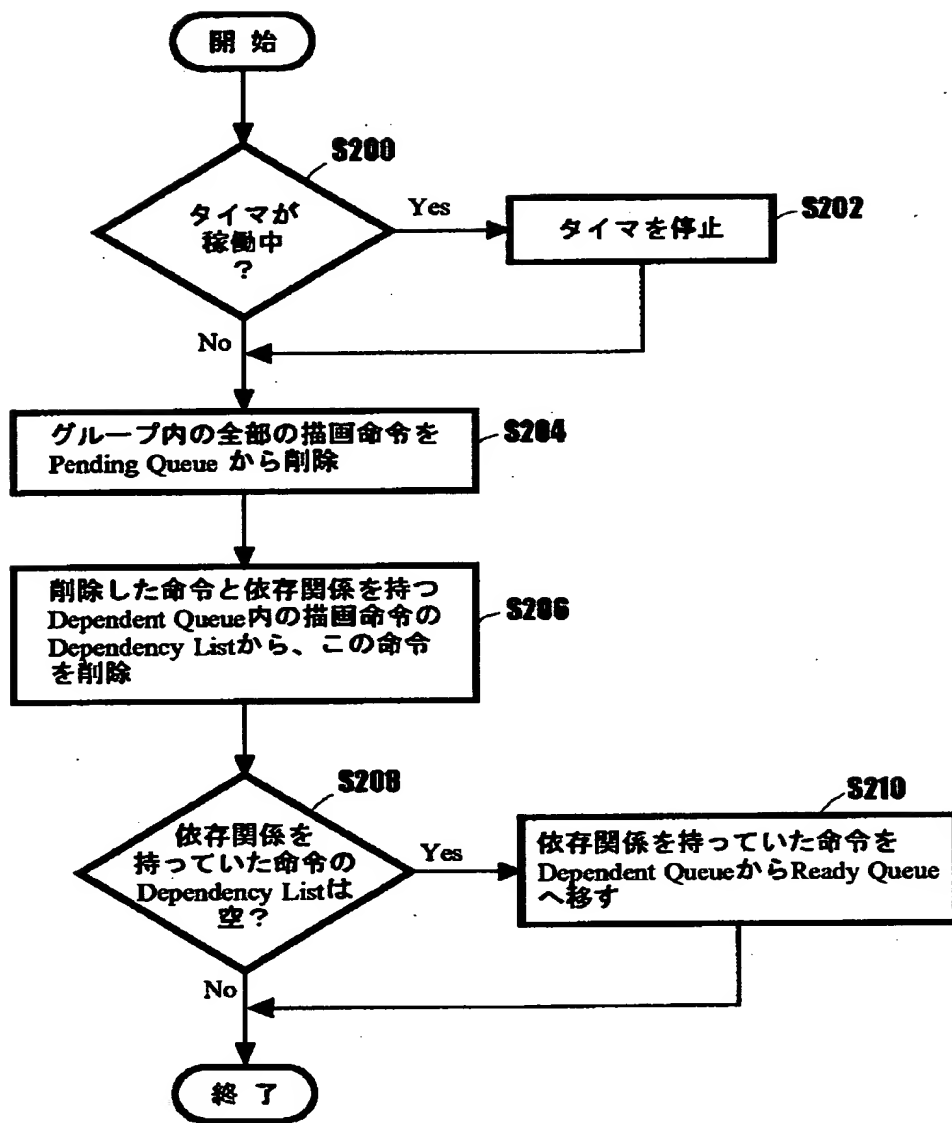


【図 1 4】



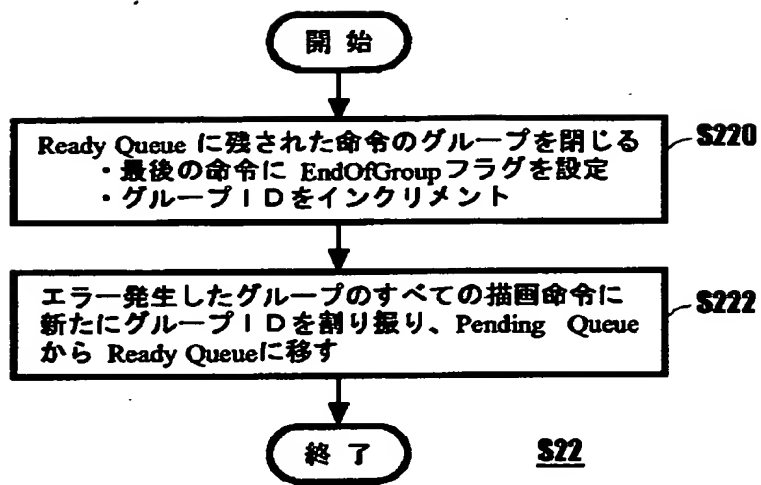
S18

【図 1 5】

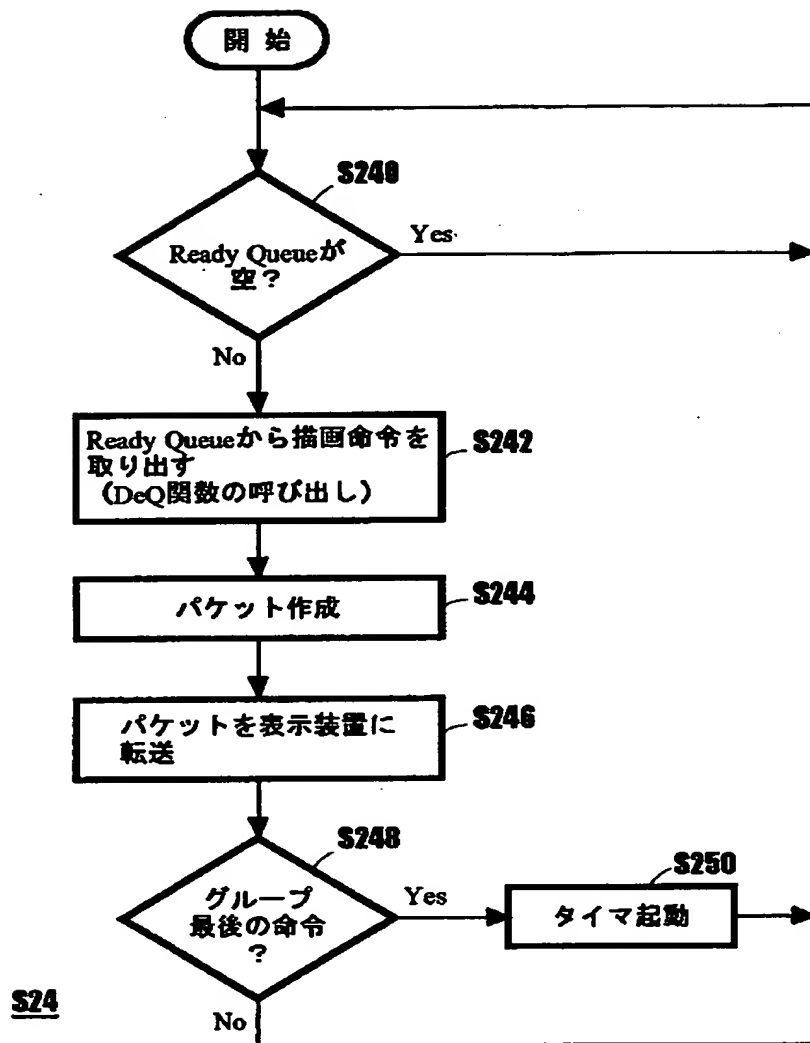


S20

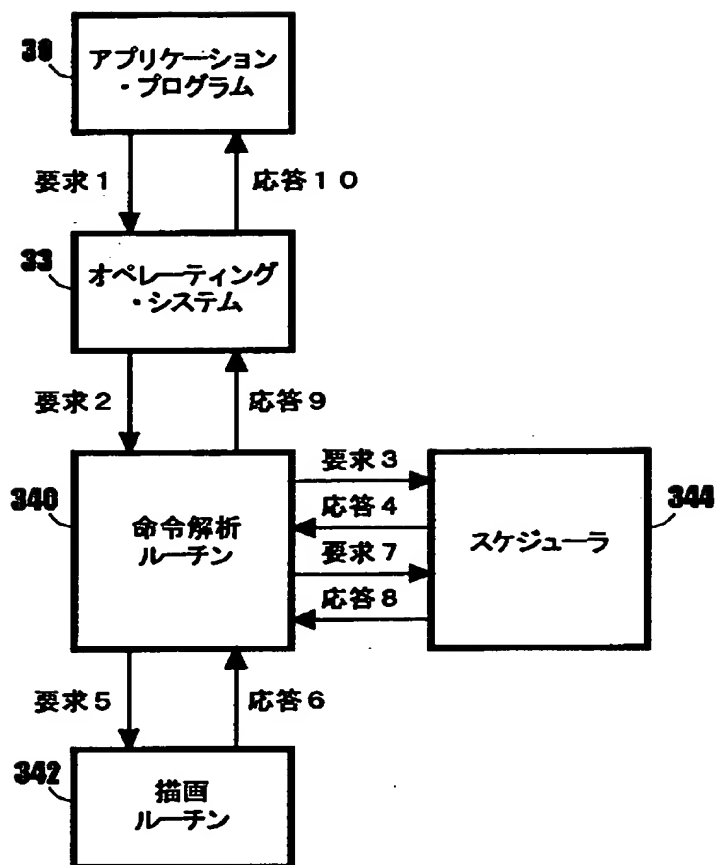
【図 1 6】



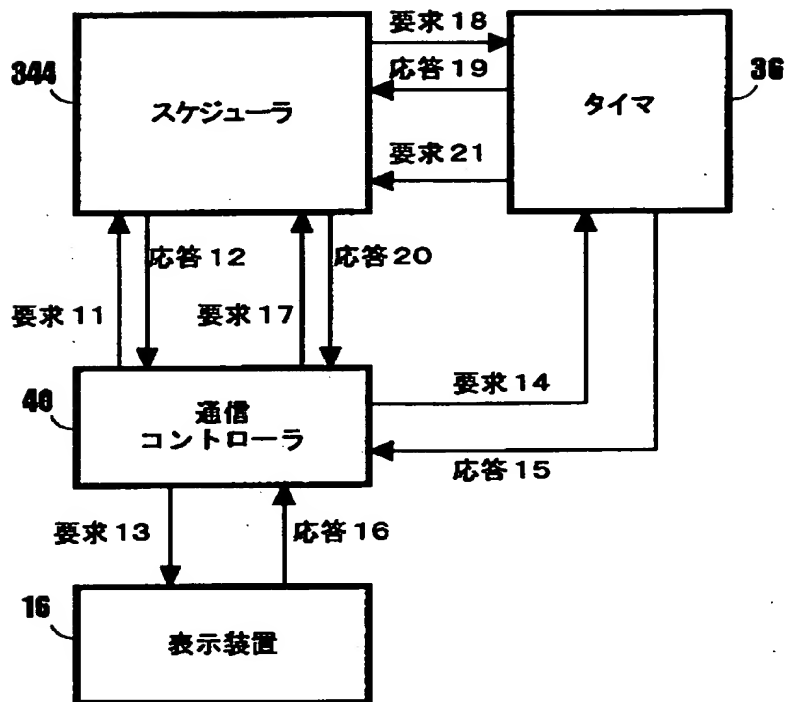
【図 1 7】



【図 1 8】



【図 1 9】



【書類名】 要約書

【要約】

【課題】 画像データを効率よくコンピュータから表示装置へ伝送する。

【解決手段】 スケジューラ 3 4 4 は、描画命令の変更により生じるデータ量増加をマージによって打ち消せるか否かを判断し、打ち消せる場合に描画命令を変更する。通信コントローラ 4 0 は、マージされた描画命令が所定のデータ量以上含まれる伝送用のパケットを生成し、表示装置に対して送信する。

【選択図】 図 6

認定・付加情報

特許出願の番号	平成11年 特許願 第287268号
受付番号	59900986544
書類名	特許願
担当官	第七担当上席 0096
作成日	平成11年10月12日

<認定情報・付加情報>

【提出日】	平成11年10月 7日
-------	-------------

出 願 人 履 歴 情 報

識別番号 [390009531]

1. 変更年月日 1990年10月24日

[変更理由] 新規登録

住 所 アメリカ合衆国10504、ニューヨーク州 アーモンク (番地なし)

氏 名 インターナショナル・ビジネス・マシーンズ・コーポレイション